AMENDMENTS TO THE CLAIMS

The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (previously presented) An apparatus in a processor for speculatively performing a return instruction, comprising:

a first call/return stack, configured for pushing thereon a plurality of return addresses of a corresponding plurality of call instructions in response to fetching from an instruction cache a plurality of cache lines predicted to include said corresponding plurality of call instructions, and for popping therefrom a first return address in response to fetching from said instruction cache a cache line predicted to include a return instruction, wherein said first return address is a top one of said plurality of return addresses simultaneously stored in said first call/return stack as a result of said pushing, wherein each of said plurality of return addresses is pushed onto said first call/return stack prior to decoding said corresponding call instruction;

a second call/return stack, configured to provide a second return address in response to decoding said return instruction, subsequent to said first call/return stack popping therefrom said first return address;

a comparator, coupled to said first and second call/return stacks, for comparing said first and second return addresses prior to the return instruction reaching an execution stage of a pipeline of the processor, wherein said execution stage is configured to finally resolve the return instruction; and

control logic, coupled to said comparator, for controlling the processor to branch to said first return address, said control logic subsequently controlling the processor to branch to said second return address if said comparator indicates said first and second return addresses do not match.

2. (previously presented) The apparatus of claim 1, wherein said second call/return stack is configured for pushing thereon a second plurality of return addresses in response to decoding said plurality of call instructions, wherein said second return address is a top one of said second plurality of return addresses simultaneously stored in said second call/return stack.

3. (previously presented) The apparatus of claim 1, further comprising an instruction buffer, coupled to said instruction cache, configured to buffer said plurality of cache lines and said cache line for provision to an instruction decoder configured to decode said plurality of call instructions and said return instruction.

4. (previously presented) The apparatus of claim 1, wherein said first call/return stack speculatively provides said first return address in response to a fetch address, said fetch address selecting said cache line fetched from said instruction cache.

5. (original) The apparatus of claim 4, wherein said first call/return stack speculatively provides said first return address in response to said fetch address whether or not said return instruction is present in said cache line.

6. (original) The apparatus of claim 1, further comprising:

a branch target address cache (BTAC), coupled to said first call/return stack, for caching a plurality of indications of whether a corresponding plurality of instructions previously executed by the processor are return instructions.

7. (original) The apparatus of claim 6, wherein said first call/return stack provides said first return address in response to said BTAC providing one of said plurality of indications, wherein said one of said plurality of indications indicates that said corresponding instruction is a return instruction.

8. (original) The apparatus of claim 7, wherein said BTAC provides said one of said plurality of indications in response to an instruction cache fetch address.

9. (original) The apparatus of claim 6, wherein said BTAC is further configured to cache a plurality of lengths of a corresponding plurality of call instructions previously executed by the processor.

10. (original) The apparatus of claim 9, wherein said first return address comprises a sum of an instruction cache fetch address and one of said plurality of lengths provided by said BTAC.

11. (original) The apparatus of claim 10, wherein said BTAC is further configured to cache a plurality of byte offsets within an instruction cache line of said corresponding plurality of call instructions, said byte offsets being within an instruction cache line selected by said fetch address.

12. (original) The apparatus of claim 11, wherein said instruction cache line is selected by said fetch address.

13. (original) The apparatus of claim 12, wherein said first return address comprises a sum of said instruction cache fetch address and said one of said plurality of lengths and one of said plurality of byte offsets.

14-18. (canceled)

19. (previously presented) A microprocessor for predicting return instruction target addresses, comprising:

an instruction cache, for generating a line of instruction bytes selected by a fetch address, said fetch address received from an address bus;

address selection logic, coupled to said address bus, for selecting said fetch address and providing said fetch address on said address bus;

a branch target address cache (BTAC), coupled to said address bus, for caching indications of previously executed return instructions and for providing one of said indications in response to said fetch address;

a first call/return stack, coupled to said BTAC, for providing a first return address to said address selection logic in response to said one of said indications, wherein said first call/return stack is configured to simultaneously store a plurality of return addresses, wherein said plurality of return addresses are pushed onto said first call/return stack in response to indications provided from said BTAC of previously executed call instructions in response to said fetch address;

decode logic, coupled to said instruction cache, for decoding said line of instruction bytes;

a second call/return stack, coupled to said decode logic, for providing a second return address to said address selection logic in response to said decode logic indicating that a return instruction is present in said line of instruction bytes, wherein said second call/return stack is configured to store a plurality of return addresses, wherein said second call/return stack is physically distinct from said first call/return stack; and

an execution stage, coupled to said decode logic, for finally resolving return instructions, wherein said first and second call/return stacks provide said first and second return addresses to said address selection logic prior to said return instruction reaching said execution stage.

20. (original) The microprocessor of claim 19, wherein said first call/return stack provides said first return address before said decode logic decodes said line of instruction bytes.

21. (original) The microprocessor of claim 19, wherein said branch target address cache provides said one of said indications in response to said fetch address whether or not a return instruction is present in said line of instruction bytes.

22. (original) The microprocessor of claim 19, wherein said first call/return stack provides said first return address in response to said one of said indications indicating said one of said previously executed return instructions is potentially present in said line of instruction bytes.

23. (original) The microprocessor of claim 19, further comprising:

control logic, coupled to said BTAC, configured to control said address selection logic to select said first return address during a first period.

24. (original) The microprocessor of claim 23, further comprising:

a comparator, coupled to said first and second call/return stacks, for comparing said first and second return addresses.

25. (original) The microprocessor of claim 24, wherein said control logic is further configured to control said address selection logic to select said second return

address subsequent to controlling said address selection logic to select said first return address if said comparator indicates said first and second return addresses do not match.

26. (original) The microprocessor of claim 19, wherein said second call/return stack provides said second return address subsequent to said first call/return stack providing said first return address.

27. (previously presented) A method for speculatively branching a microprocessor to a target address of a return instruction, the microprocessor including an execution stage for finally resolving the return instruction, the method comprising:

pushing onto a first call/return stack a plurality of return addresses of a corresponding plurality of call instructions, causing said plurality of return addresses to be simultaneously stored in said first call/return stack, wherein for each of said plurality of return addresses said pushing is performed prior to decoding of said corresponding call instruction;

generating a first target address by popping one of said plurality of return addresses off a top of said first call/return stack;

branching to said first target address;

generating a second target address by a second call/return stack subsequent to said branching to said first target address, wherein the second call/return stack is configured to store a plurality of return addresses, wherein the second call/return stack is physically distinct from the first call/return stack;

comparing said first and second target addresses prior to the return instruction reaching the execution stage; and

branching to said second target address if said first and second target addresses do not match.

28. (original) The method of claim 27, wherein said branching to said first target address comprises selecting said first target address and providing said first target address as a fetch address to an instruction cache in the microprocessor.

29. (original) The method of claim 28, wherein said generating said first target address comprises said first call/return stack generating said first target address in response to a previous fetch address that was provided to said instruction cache.

30. (original) The method of claim 29, wherein said generating said first target address is performed whether or not a return instruction is present in an instruction cache line selected by said fetch address.

31. (original) The method of claim 29, further comprising:

decoding a return instruction present in a line of instruction bytes selected from said instruction cache by said fetch address, wherein said decoding said return instruction present in said line of instruction bytes is performed subsequent to said branching to said first target address.

32. (original) The method of claim 31, wherein said generating said second target address comprises said second call/return stack generating said second target address in response to said decoding said return instruction present in said line of instruction bytes.

33. (previously presented) The method of claim 27, wherein said pushing onto said first call/return stack said plurality of return addresses is performed in response to fetching from an instruction cache a corresponding plurality of cache lines each predicted to contain at least one call instruction, wherein said pushing onto said first call/return stack is performed for each of said plurality of return addresses prior to decoding said at least one call instruction.

34. (original) The method of claim 33, further comprising:

pushing said first target address onto said first call/return stack prior to said popping said first target address off said first call/return stack.

35. (original) The method of claim 34, further comprising:

calculating said first target address prior to said pushing.

36. (original) The method of claim 35, wherein said calculating said first target address comprises adding a cached length of a previously cached call instruction and a fetch address selecting an instruction cache line potentially including said previously executed call instruction.

37. (original) The method of claim 36, wherein said generating said first target address comprises adding said fetch address, said cached length, and a cached offset of said call instruction within said instruction cache line.

38. (original) The method of claim 34, wherein said pushing is performed in response to an instruction cache fetch address.

39. (previously presented) A microprocessor for predicting return instruction target addresses, comprising:

an instruction cache, for providing a line of instructions in response to a fetch address received on an address bus;

a multiplexer, having a plurality of inputs, configured to select one of said plurality of inputs for provision on said address bus as said fetch address to said instruction cache;

a speculative branch target address cache (BTAC), coupled to said address bus, for indicating a speculative presence of a return instruction in said line of instructions;

a speculative call/return stack, coupled to said speculative BTAC, for providing a speculative return address to a first of said plurality of multiplexer inputs in response to said speculative BTAC indicating said speculative presence of said return instruction, wherein said speculative call/return stack is configured to simultaneously store a plurality of return addresses, wherein said plurality of return addresses are pushed onto said speculative

call/return stack in response to instances of said speculative BTAC indicating a speculative presence of a call instruction in said line of instructions;

decode logic, configured to receive and decode said line of instructions;

a non-speculative call/return stack, coupled to said decode logic, for providing a non-speculative return address to a second of said plurality of multiplexer inputs in response to said decode logic indicating that said return instruction is actually present in said line of instructions, wherein said speculative call/return stack is configured to store a plurality of return addresses, wherein said non-speculative call/return stack is physically distinct from said speculative call/return stack; and

a comparator, coupled to said speculative and non-speculative call/return stacks, for comparing said speculative and non-speculative return addresses prior to said return instruction reaching an execution stage of a pipeline of the processor, wherein said execution stage is configured to finally resolve the return instruction;

wherein said multiplexer selects said speculative return address in a first instance, and selects said non-speculative return address in a second instance subsequent to said first instance if said comparator indicates that said speculative and non-speculative return addresses do not match.

40. (previously presented) A method for predicting a return address of a return instruction in a microprocessor, the method comprising:

pushing a first return address onto a first call/return stack, in response to fetching from an instruction cache a first cache line predicted to include a first call instruction;

pushing a second return address onto the first call/return stack, in response to fetching from the instruction cache a second cache line predicted to include a second call instruction;

popping the second return address from the first call/return stack, in response to fetching from the instruction cache a cache line predicted to include a first return instruction;

branching the microprocessor to the second return address, after said popping the second return address;

popping the first return address from the first call/return stack, in response to fetching from the instruction cache a cache line predicted to include a second return instruction;

branching the microprocessor to the first return address, after said popping the first return address;

pushing a third return address onto a second call/return stack, in response to decoding the first call instruction, after said popping the first return address;

pushing a fourth return address onto the second call/return stack, in response to decoding the second call instruction;

popping the fourth return address from the second call/return stack, in response to decoding the first return instruction;

comparing the second and fourth return addresses prior to the first return instruction reaching an execution stage of a pipeline of the processor, wherein the execution stage is configured to finally resolve the first return instruction; and

branching the microprocessor to the fourth return address, after said popping the fourth return address, if the second and fourth return addresses do not match.

41. (previously presented) A branch prediction apparatus in a processor, comprising:

a first call/return stack, configured for:

pushing thereon a first return address, in response to fetching from an instruction cache a first cache line predicted to include a first call instruction;

pushing thereon a second return address, in response to fetching from the instruction cache a second cache line predicted to include a second call instruction; and

popping therefrom the second return address, in response to fetching from the instruction cache a cache line predicted to include a first return instruction;

control logic, coupled to said first call/return stack, configured to branch the microprocessor to the first return address, after said popping the first return address;

wherein said first call/return stack is further configured for popping therefrom the first return address, in response to fetching from the instruction cache a cache line predicted to include a second return instruction;

wherein said control logic is further configured to branch the microprocessor to the first return address, after said popping the first return address;

a second call/return stack, configured for:

pushing thereon a third return address, in response to decoding the first call instruction, after said popping the first return address;

pushing thereon a fourth return address, in response to decoding the second call instruction; and

popping therefrom the fourth return address, in response to decoding the first return instruction;

a comparator, coupled to said first and second call/return stacks, configured to compare the second and fourth return addresses prior to the first return instruction reaching an execution stage of a pipeline of the processor, wherein the execution stage is configured to finally resolve the first return instruction; and

wherein said control logic is further configured to branch the microprocessor to the fourth return address, after said popping the fourth return address, if the second and fourth return addresses do not match.